

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE  
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

In re Patent Application of	)	
Sanjay GHEMAWAT et al.	)	<b>M/S: Appeal Brief - Patents</b>
	)	
Application No.: 10/608,039	)	Group Art Unit: 2161
	)	
Filed: June 30, 2003	)	Examiner: C. Daye
	)	
For: GARBAGE COLLECTING	)	
SYSTEMS AND METHODS	)	

U.S. Patent and Trademark Office  
Customer Window, Mail Stop Appeal Brief - Patents  
Randolph Building  
401 Dulany Street  
Alexandria, VA 22314

**REPLY BRIEF UNDER 37 C.F.R. § 41.41**

This Reply Brief is submitted in response to the Examiner's Answer, mailed July 23,  
2007.

I. STATUS OF CLAIMS

Claims 1-25 are pending in this application.

Claims 1, 2, 4-7, 10-13, and 20-25 have been finally rejected under 35 U.S.C. § 102(b) as anticipated by Mattis et al. (U.S. Patent No. 6,209,003).

Claim 3 has been finally rejected under 35 U.S.C. § 103(a) as unpatentable over Mattis et al. in view of Manley et al. (U.S. Patent Application Publication No. 2003/0182330).

Claims 8, 9, and 14-19 have been finally rejected under 35 U.S.C. § 103(a) as unpatentable over Mattis et al. in view of Hisgen et al. ("New-Value Logging in the Echo Replicated File System," June 23, 1993).

II. GROUND OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1, 2, 4-7, 10-13, and 20-25 stand rejected under 35 U.S.C. § 102(b) as anticipated by Mattis et al.

B. Claim 3 stands rejected under 35 U.S.C. § 103(a) as unpatentable over Mattis et al. in view of Manley et al.

C. Claims 8, 9, and 14-19 stand rejected under 35 U.S.C. § 103(a) as unpatentable over Mattis et al. in view of Hisgen et al.

III. RESPONSE TO ARGUMENT SECTION OF EXAMINER'S ANSWERA. **The Rejection Under 35 U.S.C. § 102(b) Based on Mattis et al. (U.S. Patent No. 6,209,003) Should be Reversed.**

## 1. Claims 1, 2, 6, and 7.

Independent claim 1 recites, among other things, renaming a file that is identified to be deleted. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest this feature of claim 1. Appeal Brief, pages 7-9.

In the Examiner's Answer, the Examiner alleged:

Mattis discloses at column 23, lines 15-23, wherein "If the fragment is to be deleted, then in step 812 it is deleted from the arena by marking it as deleted and overwriting the data in the fragment. When an object 52 is stored in multiple fragments, and the garbage collection process determines that one of the fragments is to be deleted, then the process deletes all fragments associated with the object". The 'marking' of the fragment as deleted is a representation of renaming a file and since it is marked as deleted, that therefore makes it the identified file to be deleted.

Examiner's Answer, page 11. Appellants submit that the Examiner's allegation lacks merit.

Marking a fragment as deleted and overwriting the data in the fragment is completely different from renaming a file that is identified to be deleted, as required by claim 1.

The Examiner also cites a portion of Appellants' application at paragraph 0071, which states that "[m]aster 130 may, however, actually only rename the file with a deletion timestamp," that allegedly supports the representation that marking a file is a form of renaming. Examiner's Answer, page 11. Appellants submit that the Examiner's allegation lacks merit. In paragraph 0071, Appellants disclose that a file to be deleted can be renamed using a deletion timestamp. In other words, the name of the file can be replaced or supplemented with a deletion timestamp. In either situation, the file is renamed. There is nothing in this section that supports the Examiner's allegation that marking a fragment as deleted and overwriting the data in the fragment is the

same as renaming a file, as required by claim 1. Thus, the Examiner's allegation lacks merit.

The Examiner also cited column 32, lines 29-37, of Mattis et al. for allegedly showing that the "setting of the deletion flag and as the deletion flag is set, the block being marked as deleted, is a further disclosure of renaming the file to be deleted." Examiner's Answer, page 11. Appellants submit that this section of Mattis et al. provides absolutely no support for the Examiner's allegation that setting a deletion flag is equivalent to renaming a file to be deleted, as recited in claim 1.

At column 32, lines 29-37, Mattis et al. discloses:

To accomplish removal of a block found in the cache, however, in step 960 the process sets the deletion flag, and checks the block in with the deletion flag set. As described herein in connection with the check-in process (steps 938 and 944 of FIG. 9B), when the deletion flag is set, the block will be marked as deleted. Thereafter, the block is eventually removed from the Directory Index when the changes reflected in the Open Directory are synchronized to the Directory Index.

In this section, Mattis et al. discloses that to remove a block, a deletion flag is set, the block is marked as deleted, and the block is eventually removed from the Directory Index. Contrary to the Examiner's allegation, nowhere in this section, or elsewhere, does Mattis et al. disclose or suggest that setting a deletion flag is equivalent to renaming a file that is identified to be deleted, as required by claim 1.

In the Appeal Brief, Appellants explained that a section of Mattis et al. relied upon by the Examiner (namely, column 3, lines 13-19) actually teaches away from renaming a file that is identified to be deleted, as required by claim 1. In particular, this section of Mattis et al. discloses that file systems, which include a prior art approach to structure cache object stores, include support for renaming files. This portion specifically states that renaming files is "irrelevant to cache object stores" and "counter-productive to [the Mattis et al.] application."

Column 3, lines 13-19. In other words, this section of Mattis et al. specifically teaches away from renaming files. This section of Mattis et al. provides direct evidence to contravene the Examiner's allegation that marking a fragment as deleted and overwriting the data in the fragment is equivalent to renaming a file that is identified to be deleted, as required by claim 1.

In the Examiner's Answer, the Examiner attempted to address these arguments by explaining that column 3, lines 13-19, was cited for the purpose of revealing that renaming of files is a well-known process within the art for garbage collection. Examiner's Answer, pages 11-12. The Examiner did not address Appellants' arguments, however, that Mattis et al. actually teaches away from renaming files. The Examiner also did not provide a single reason why one of ordinary skill in the art at the time of Appellants' invention would have sought to modify the Mattis et al. system to include renaming files. Thus, the Examiner did not establish a proper rejection under 35 U.S.C. § 102.

Independent claim 1 further recites, among other things, permanently deleting the renamed file a predetermined amount of time after renaming the identified file as part of a garbage collection process. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest this feature of claim 1. Appeal Brief, pages 9-14.

In the Examiner's Answer, the Examiner alleged:

To begin, as background information, well known within the art, garbage collection is a form of automatic memory management. Garbage collection reclaims garbage, or memory used by objects that will never again be accessed or mutated by the application. As such, the Mattis reference is "a method for garbage collection in a cache of information objects. . .the garbage collection periodically selects a pool that is storing an amount of data greater than a minimum storage value or high water mark. Each arena in the pool is examiner and selected for garbage collection according to selection criteria. Each fragment within a selected arena is examiner based upon a second set of selection criteria that determine whether the fragment is retained or deleted' (See ABSTRACT), therefore, the reference as a whole is part of a garbage collection process.

Examiner's Answer, page 12. Without acquiescing in the Examiner's definition of garbage collection, Appellants submit that the Examiner has only shown that Mattis et al. is directed to garbage collection, not that Mattis et al. discloses or suggests permanently deleting a renamed file a predetermined amount of time after renaming the identified file as part of a garbage collection process, as required by claim 1.

In the Examiner's Answer, the Examiner continued to rely on column 32, lines 29-37, of Mattis et al. for allegedly disclosing permanently deleting a renamed file a predetermined amount of time after renaming the identified file as part of a garbage collection process, as recited in claim 1. Examiner's Answer, page 12. Appellants addressed column 32, lines 29-37, of Mattis et al. at page 12 of the Appeal Brief.

At page 12 of the Examiner's Answer, the Examiner newly relied upon column 35, lines 5-20, and column 35, line 65 - column 36, line 7 of Mattis et al. for allegedly disclosing permanently deleting a renamed file a predetermined amount of time after renaming the identified file as part of a garbage collection process, as recited in claim 1. Appellants submit that these sections of Mattis et al. provide no support for the Examiner's allegation.

At column 35, lines 5-20, Mattis et al. discloses:

Accordingly, in step 882 the process tests whether the matching block has been marked as deleted, and currently has no other processes reading it or writing it. If the values of both the reader counter and the writer counter are zero, then the block has no other processes reading it or writing it. If the values of either the reader counter or the writer counter are nonzero, or if the matching block has not been marked as deleted, then the block is a valid previously existing block that cannot be created. In step 884 an error message is returned to the protocol engine 70 or cache 80 indicating that the current block is not available to be created.

If the matching block is deleted and has no writers or readers accessing it, then the process can effectively create a new block by clearing and initializing the matching, previously created block.

In this section, Mattis et al. discloses that if a block has been marked as deleted and currently has no other processes reading it or writing it, then a new block can be created by clearing and initializing the block. Contrary to the Examiner's allegation, nowhere in this section, or elsewhere, does Mattis et al. disclose or suggest permanently deleting a renamed file a predetermined amount of time after renaming a file that is identified to be deleted as part of a garbage collection process, as required by claim 1.

At column 35, line 65 - column 36, line 7, Mattis et al. discloses:

In this way, the Open Directory is updated with any changes that were carried out by the process that modified the copy of the block that was obtained in the checkout process. Thereafter, and if the test of step 936 is negative, the process tests whether a delete check-in flag is set. The delete check-in flag indicates that the block is to be deleted after check-in. The delete flag is an argument to the checkin operation. If the flag is set, then in step 944 the process marks the block as deleted. Processing concludes at step 940.

In this section, Mattis et al. discloses a delete check-in flag that indicates that a block is to be deleted after a check-in process. Contrary to the Examiner's allegation, nowhere in this section, or elsewhere, does Mattis et al. disclose or suggest permanently deleting a renamed file a predetermined amount of time after renaming a file that is identified to be deleted as part of a garbage collection process, as required by claim 1.

In the Examiner's Answer, the Examiner alleged that the above-identified sections of Mattis et al. show:

The setting of the deletion flag and in connection with the check-in process, marking the block as deleted, and eventually removing the block fully discloses the permanently deleting the renamed file a predetermined amount of time after renaming the file. Also, because the delete check-in flag indicates that the block is to be deleted after check-in, which is a predetermined amount of time after renaming the file, further discloses the above-argued limitation.

Examiner's Answer, page 13. Appellants submit that there is no merit to the Examiner's

allegations. As explained above and in the Appeal Brief, Mattis et al. does not disclose or suggest renaming a file that is to be deleted. Therefore, contrary to the Examiner's allegation, Mattis et al. does not disclose or suggest permanently deleting a renamed file a predetermined amount of time after renaming a file that is identified to be deleted as part of a garbage collection process, as required by claim 1.

Even assuming, for the sake of argument, that buffering changes to Directory Table 110, setting a deletion flag for a block, or setting a check-in flag for a block, as disclosed by Mattis et al., can reasonably be interpreted to correspond to renaming a file, as alleged by the Examiner, Mattis et al. does not disclose doing anything a predetermined amount of time after performing any of these actions. In fact, the Examiner has only established that Mattis et al. discloses performing garbage collection when the amount of active storage in the pool becomes greater than a high water mark. Mattis et al. at column 22, lines 14-23. Thus, Mattis et al. does not disclose or suggest permanently deleting a renamed file a predetermined amount of time after renaming a file that is identified to be deleted as part of a garbage collection process, as required by claim 1.

Independent claim 1 further recites, among other things, identifying, to one of the servers, one of the chunks that corresponds to the permanently deleted file. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest this feature of claim 1. Appeal Brief, pages 14-15.

In the Examiner's Answer, the Examiner newly relied upon column 23, lines 15-37 of Mattis et al. for allegedly disclosing identifying, to one of the servers, one of the chunks that corresponds to the permanently deleted file. Examiner's Answer, page 13. Appellants submit

that this section of Mattis et al. provides no support for the Examiner's allegation.

At column 23, lines 15-37, Mattis et al. discloses:

If the fragment is to be deleted, then in step 812 it is deleted from the arena by marking it as deleted and overwriting the data in the fragment. When an object 52 is stored in multiple fragments, and the garbage collection process determines that one of the fragments is to be deleted, then the process deletes all fragments associated with the object. This may involve following a chain of fragments, of the type shown in FIG. 5, to another arena or even another pool.

If the fragment is not to be deleted, then in step 810 the fragment is written to a new arena. FIG. 8B, which is discussed below, shows preferred sub-steps involved in carrying out step 810.

After the fragment is deleted or moved to another arena, in step 814 the Directory Table 110 is updated to reflect the new location of the fragment. Step 814 involves using the value of the key 206a in the fragment header 208d associated with a fragment 208n to be updated to look up a block 112a-112n that is associated with the fragment. When the correct Directory Table block 112a-112n is identified, the disk location value 118 in the block is updated to reflect the new location of the fragment. If the fragment has been deleted, then any corresponding Directory Table entries are deleted.

In this section, Mattis et al. discloses that a fragment to be deleted from the arena is marked as deleted and the data of the fragment is overwritten. A fragment that is not to be deleted is written to a new arena. Nowhere in this section, or elsewhere, does Mattis et al. disclose or suggest identifying, to one of a plurality of servers that stores files as chunks, one of the chunks that corresponds to the permanently deleted file, as required by claim 1.

The Examiner alleged that the above-identified section of Mattis et al. discloses:

Since the object is in multiple fragments and the garbage collection process determines that one of the fragments is to be deleted, then all of the fragments associated are affected as well. Also, any corresponding fragments are identified and deleted as well. This information is identified to the server, as understood at column 7, lines 23-29.

Examiner's Answer, page 14. Appellants submit that the cited section of Mattis et al. (i.e., column 7, lines 23-29) provides no support for the Examiner's allegation.

At column 7, lines 23-29, Mattis et al. discloses:

In this context, the term "object" means a network resource or any discrete element of information that is delivered from a server. Examples of objects include Web pages or documents, graphic images, files, text documents, and objects created by Web application programs during execution of the programs, or other elements stored on a server that is accessible through the Internet 20.

In this section, Mattis et al. discloses the definition of the term "object." Contrary to the Examiner's allegation, simply because an object is defined as a discrete element of information that is delivered from a server, does not mean or remotely suggest that one of the chunks that corresponds to a permanently deleted file is identified to one of a plurality of servers that stores files as chunks, as required by claim 1.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claims 1, 2, 6, and 7 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claims 1, 2, 6, and 7 be reversed.

2. Claim 4.

Dependent claim 4 recites that the predetermined amount of time is a user-configurable amount of time. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest this feature of claim 4. Appeal Brief, pages 15-17.

In the Examiner's Answer, the Examiner alleged that column 22, lines 8-23, of Mattis et al. discloses:

watermarks (both high and low) wherein when an active storage is above the appropriate capacity, then the system is made aware and garbage collection is carried out in order to decrease the capacity back down to the low water mark level. As discussed above, the predetermined amount of time is associated with the permanent deletion of the renamed file during the garbage collection process. As such, the water marks being relied upon have been programmed within the system (by a user) as a way of determining when the storage capacity meets or exceeds the correct amount, making the amount of time user-configurable.

Examiner's Answer, pages 14-15. As explained above with regard to claim 1, Mattis et al. does not disclose or suggest permanently deleting a renamed file a predetermined amount of time after renaming the file. Therefore, even assuming, for the sake of argument, that a user in Mattis et al. programs the high and low water marks, Mattis et al. does not disclose or remotely suggest permanently deleting a renamed file a user-configurable amount of time after renaming the file, as required by claim 4.

In the Examiner's Answer, the Examiner also newly cited column 36, lines 1-7, of Mattis et al. for allegedly disclosing permanently deleting a renamed file a user-configurable amount of time after renaming the file, as recited in claim 4. Examiner's Answer, page 15. Appellants disagree.

At column 36, lines 1-7, Mattis et al. discloses:

Thereafter, and if the test of step 936 is negative, the process tests whether a delete check-in flag is set. The delete check-in flag indicates that the block is to be deleted after check-in. The delete flag is an argument to the checkin operation. If the flag is set, then in step 944 the process marks the block as deleted. Processing concludes at step 940.

In this section, Mattis et al. discloses testing a delete check-in flag and, if the delete check-in flag indicates that the block is to be deleted, then the block is marked as deleted. The Examiner alleged that:

because the delete check-in flag indicates that the block is to be deleted after check-in, the program has been programmed or told to do so by the user, thereby making the amount of time user-configurable.

Examiner's Answer, page 15. Appellants submit that the Examiner's conclusion is based solely on impermissible hindsight reasoning in a flawed attempt to reconstruct the claimed invention. Nowhere in the above-identified section, or elsewhere, does Mattis et al. disclose or suggest permanently deleting a renamed file a user-configurable amount of time after renaming the file,

as required by claim 4.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 4 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claim 4 be reversed.

3. Claim 5.

Dependent claim 5 recites that the user-configurable amount of time differs for different ones of the files. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest this feature of claim 5. Appeal Brief, pages 17-18.

In the Examiner's Answer, the Examiner newly cited to column 35, lines 45-48, and column 36, lines 1-7, of Mattis et al. for allegedly disclosing the feature of claim 5. Examiner's Answer, page 16. Appellants submit that the disclosure of Mattis et al. provides no support for the Examiner's allegation.

At column 35, lines 45-48, Mattis et al. discloses:

FIG. 9B is a flow diagram of a block check-in process. The cache 80 carries out the process of FIG. 9B to check a block into the Open Directory 130 after the block is read, modified, or deleted.

In this section, Mattis et al. discloses a block check-in process that is carried out after the block is read, modified, or deleted. Contrary to the Examiner's allegation, nothing in this section discloses or remotely suggests permanently deleting a renamed file a user-configurable amount of time after renaming the file, where the user-configurable amount of time differs for different ones of the files, as required by claim 5.

Column 36, lines 1-7, of Mattis et al. is reproduced above. In this section, Mattis et al. discloses testing a delete check-in flag and, if the delete check-in flag indicates that the block is

to be deleted, then the block is marked as deleted. Contrary to the Examiner's allegation, nothing in this section discloses or remotely suggests permanently deleting a renamed file a user-configurable amount of time after renaming the file, where the user-configurable amount of time differs for different ones of the files, as required by claim 5.

The Examiner alleged that the above-identified sections of Mattis et al. disclose:

once the block has been checked in, and if the block has a delete check-in flag, the system then deletes the block after the check-in. Since the check-in for a file can differ dependent upon the size and amount of information within the block, and also since the information can only be processed one at a time, therefore affects each file differently as far as when the user-configurable amount of time will actually permanently delete the renamed file.

Examiner's Answer, page 16. The Examiner's reasoning is based solely on impermissible hindsight in a flawed attempt to reconstruct the claimed invention. The Examiner has not established that Mattis et al. discloses or remotely suggests permanently deleting a renamed file a user-configurable amount of time after renaming the file, where the user-configurable amount of time differs for different ones of the files, as required by claim 5. Therefore, the Examiner has not established a proper rejection under 35 U.S.C. § 102.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 5 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claim 5 be reversed.

4. Claims 10 and 11.

Dependent claim 10 recites maintaining versions of the chunks; identifying a stale chunk based on the versions of the chunks; and deleting the stale chunk. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest these features of claim 10. Appeal Brief, pages 19-20.

In the Examiner's Answer, the Examiner newly cited column 17, lines 39-46, and column 26, lines 8-22, of Mattis et al. for allegedly disclosing identifying a stale chunk based on the version of the chunks. Examiner's Answer, page 17. Appellants submit that the disclosure of Mattis et al. does not support the Examiner's allegation.

At column 17, lines 39-46, Mattis et al. discloses:

Each pool header 202 stores a Magic number, a Version No. value, a No. of Arenas value, and one or more arena headers 206a-206n. The Magic number is used solely for internal consistency checks. The Version No. value stores a version number of the program or process that created the arenas 206a-206n in the pool. It is used for consistency checks to ensure that the currently executing version of the cache 80 can properly read and write the arenas.

In this section, Mattis et al. discloses a pool header that stores a version number of the program or process that created the arenas in the pool. Nowhere in this section, or elsewhere, does Mattis et al. disclose or suggest maintaining versions of chunks, let alone identifying a stale chunk based on the versions of the chunks, as required by claim 10. Instead, Mattis et al. simply discloses storing a version number of a program or process.

At column 26, lines 8-22, Mattis et al. discloses:

If the Read Counter value is high, then the information object has been loaded recently. In that case, in block 1210 the cache sends a positive response message to the requesting process. Otherwise, as indicated in block 1212, the information object has not been loaded recently. Accordingly, as shown in block 1214, the cache sends a negative responsive message to the calling program or process. In block 1216, the cache updates an expiration date value stored in association with the information object to reflect the current date or time. By updating the expiration date, the cache ensures that the garbage collection process will not delete the object, because after the update it is not considered old. In this way, an old object is refreshed in the cache without retrieving the object from its origin, writing it in the cache, and deleting a stale copy of the object.

In this section, Mattis et al. discloses using a read counter to indicate whether an information object has been loaded recently. Nowhere in this section, or elsewhere, does Mattis et al. disclose or suggest maintaining versions of chunks, let alone identifying a stale chunk based on

the versions of the chunks, as required by claim 10. Instead, Mattis et al. simply uses a high read counter value to reflect an information object that has been recently loaded.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claims 10 and 11 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claims 10 and 11 be reversed.

5. Claim 12.

Independent claim 12 recites, among other things, means for permanently deleting a file during a garbage collection process that occurs after logging deletion of the file. In the Appeal Brief, Appellants noted that the Examiner had never addressed this feature of claim 12. Appeal Brief, pages 20-21.

In the Examiner's Answer, the Examiner newly cited to several sections of Mattis et al. that allegedly disclose means for permanently deleting a file during a garbage collection process that occurs after logging deletion of the file, as recited in claim 12. Examiner's Answer, pages 18-19.

Without acquiescing in the Examiner's allegations, Appellants submit that claim 12 is not anticipated by Mattis et al. for the reasons given in the Appeal Brief. Appeal Brief, pages 20-21.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 12 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claim 12 be reversed.

6. Claim 13.

Independent claim 13 recites, among other things, a master connected to the servers and

configured to identify one of the files to be deleted, rename the identified file, permanently delete one or more chunks associated with the renamed file a predetermined amount of time after renaming the identified file as part of a garbage collection process, receive, from the servers, information concerning chunks stored by the servers, and identify, to one of the servers, one of the chunks that corresponds to one of the one or more permanently deleted chunks. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest these features of claim 13. Appeal Brief, pages 21-22.

In the Examiner's Answer, the Examiner alleged that computer system 1100 in Fig. 11 is equivalent to the master recited in claim 13. Examiner's Answer, pages 19-20. Even assuming, for the sake of argument, that computer system 1100 can reasonably be equated to a master, Mattis et al. does not disclose or remotely suggest that computer system 1100 is configured to rename a file that is identified to be deleted; permanently delete one or more chunks, stored by a plurality of servers, associated with the renamed file a predetermined amount of time after renaming the identified file as part of a garbage collection process; or identify, to one of the servers, one of the chunks that corresponds to one of the one or more permanently deleted chunks, as required by claim 13.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 13 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claim 13 be reversed.

7. Claims 20 and 22-25.

Independent claim 20 recites associating version information with replicas of chunks; identifying stale replicas based on the associated version information; deleting the stale replicas;

receiving, from the servers, information concerning replicas stored by the servers; and identifying, to one of the servers, one of the replicas that corresponds to one of the deleted stale replicas. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. does not disclose or suggest these feature of claim 20. Appeal Brief, pages 23-24.

In the Examiner's Answer, the Examiner alleged that Mattis et al. discloses that "object replicas are referred to as a cache." Examiner's Answer, pages 20-21. Appellants submit that the Examiner's allegation is incorrect. Mattis et al. discloses that "[e]ach local repository for object replicas is generally referred to as a cache" (emphasis added). Column 1, lines 61-62. Thus, the Examiner has misinterpreted the disclosure of Mattis et al.

In the Examiner's Answer, the Examiner also newly cited column 17, lines 39-46, and column 26, lines 8-22, of Mattis et al. for allegedly disclosing identifying stale replicas based on associated version information. Examiner's Answer, pages 21-22. Appellants submit that the disclosure of Mattis et al. does not support the Examiner's allegation for at least reasons similar to reasons given above with regard to claim 10.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claims 20 and 22-25 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claims 20 and 22-25 be reversed.

8. Claim 21.

Dependent claim 21 recites that the version information for one of the replicas is updated each time a lease is granted for the one of the replicas. In the Appeal Brief, Appellants provided

substantial reasons why Mattis et al. does not disclose or suggest this feature of claim 21.

Appeal Brief, pages 24-26.

In the Examiner's Answer, the Examiner alleged that Mattis et al., at column 26, lines 8-20, discloses:

updating of an expiration date to reflect the current date or time, and by updating the information, the cache ensuring the garbage collection process will not delete the object because it is not old anymore, corresponds to the updating each time a lease is granted.

Examiner's Answer, page 23. Appellants submit that the Examiner's allegation is flawed. Mattis et al. does not disclose or suggest a lease for a replica. Therefore, Mattis et al. cannot disclose or suggest version information for one of the replicas that is updated each time a lease is granted for the one of the replicas, as required by claim 21. Instead, Mattis et al. simply discloses updating an expiration date value when an information object has not been loaded recently. Figure 12; and column 26, lines 8-17.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 21 under 35 U.S.C. § 102(b) based on Mattis et al. is improper. Accordingly, Appellants request that the rejection of claim 21 be reversed.

**B. The Rejection Under 35 U.S.C. § 103(a) Based on Mattis et al. (U.S. Patent No. 6,209,003) in View of Manley et al. (U.S. Patent Application Publication No. 2003/0182330) Should be Reversed.**

**1. Claim 3.**

Dependent claim 3 recites receiving an un-deletion instruction regarding the file, and restoring an original name to the file without permanently deleting the renamed file. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. and Manley et al.,

whether taken alone or in any reasonable combination, do not disclose or suggest these features of claim 3. Appeal Brief, pages 27-30.

In the Examiner's Answer, the Examiner alleged that at paragraph 0067, Manley et al. discloses:

The file data block, which contains a pointer to the original inode, allows the system to point to new blocks while still retaining the old (i.e., original) block. Therefore, allowing the original name of the renamed file to be restored.

Examiner's Answer, pages 23-24. Appellants submit that the Examiner's allegation that using a pointer to point to new blocks while still retaining the old block is equivalent to restoring an original name to a file without permanently deleting the renamed file lacks merit and is based solely on impermissible hindsight in a flawed attempted to reconstruct the claimed invention.

In the Examiner's Answer, the Examiner also alleged that at paragraph 0118, Manley et al. discloses:

Since the file that has been marked to be deleted was given a hard link to the original file, represented by the specific entry, so the path to the data remains. Once the file is undeleted, the file survives deletion and is given its original name because of the original linkage.

Examiner's Answer, pages 24-25. Appellants submit that the Examiner's allegation that using a hard link between a file in a snapshot directory and an entry in a purgatory directory is equivalent to restoring an original name to a file without permanently deleting the renamed file lacks merit and is based solely on impermissible hindsight in a flawed attempted to reconstruct the claimed invention.

In the Examiner's Answer, the Examiner further alleged that at paragraph 0132, Manley et al. discloses:

As such, because of the undoing of the set of changes and the delay in unlinking files

until the end of the transfer, allows the rollback to resurrect files bringing them back to the original status.

Examiner's Answer, page 25. Appellants submit that the Examiner's allegation that undoing a set of changes is equivalent to restoring an original name to a file without permanently deleting the renamed file lacks merit and is based solely on impermissible hindsight in a flawed attempt to reconstruct the claimed invention.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 3 under 35 U.S.C. § 103(a) based on Mattis et al. and Manley et al., whether taken alone or in any reasonable combination, is improper. Accordingly, Appellants request that the rejection of claim 3 be reversed.

**C. The Rejection Under 35 U.S.C. § 103(a) Based on Mattis et al. (U.S. Patent No. 6,209,003) in View of Hisgen et al. ("New-Value Logging in the Echo Replicated File System," June 23, 1993) Should be Reversed.**

1. Claims 14, 15, 18, and 19.

Independent claim 14 recites, among other things, identifying, to one of the servers, one of the chunks that corresponds to one of the deleted orphaned chunks. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. and Hisgen et al. do not disclose or suggest this feature of claim 14. Appeal Brief, pages 32-33.

In the Examiner's Answer, the Examiner alleged that Hisgen et al., at page 24, paragraphs 3 and 4, discloses:

The list of the orphan files corresponds to identifying of the chunks that correspond to the orphaned chunks. Also, the EchoBox can remove the file from its orphan list and delete the file, which corresponds to the deletion of the orphaned chunks.

Examiner's Answer, page 26. Even assuming, for the sake of argument, that Hisgen et al.

discloses a list of orphan files, Appellants submit that the Examiner still has not established that Hisgen et al. discloses or suggests identifying, to one of the servers from which information concerning chunks stored by the servers was received, one of the chunks that corresponds to one of the deleted orphaned chunks, as required by claim 14.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claims 14, 15, 18, and 19 under 35 U.S.C. § 103(a) based on Mattis et al. and Hisgen et al., whether taken alone or in any reasonable combination, is improper. Accordingly, Appellants request that the rejection of claims 14, 15, 18, and 19 be reversed.

3. Claim 16.

Dependent claim 16 recites deleting, by the one of the servers, the one of the chunks that corresponds to one of the orphaned chunks. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. and Hisgen et al. do not disclose or suggest this feature of claim 16. Appeal Brief, pages 33-34.

In the Examiner's Answer, the Examiner alleged that Hisgen et al., at page 24, paragraphs 3 and 4, discloses:

The list of the orphan files corresponds to identifying of the chunks that correspond to the orphaned chunks. Also, the EchoBox can remove the file from its orphan list and delete the file, which corresponds to the deletion of the orphaned chunks.

Examiner's Answer, page 26. Even assuming, for the sake of argument, that Hisgen et al. discloses a list of orphan files, removing a file from its orphan list, and deleting the file, Appellants submit that the Examiner still has not established that Hisgen et al. discloses or suggests deleting, by the one of the servers, the one of the chunks that corresponds to one of the

orphaned chunks, as required by claim 16.

Further, the Examiner still has not provided any motivation for combining this alleged feature of Hisgen et al. with the system of Mattis et al. Therefore, the Examiner has not established a proper rejection under 35 U.S.C. § 103.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief, Appellants submit that the rejection of claim 16 under 35 U.S.C. § 103(a) based on Mattis et al. and Hisgen et al., whether taken alone or in any reasonable combination, is improper. Accordingly, Appellants request that the rejection of claim 16 be reversed.

4. Claim 17.

Dependent claim 17 recites that the deletion of the orphaned chunks occurs as part of a garbage collection process. In the Appeal Brief, Appellants provided substantial reasons why Mattis et al. and Hisgen et al. do not disclose or suggest this feature of claim 17. Appeal Brief, pages 34-35.

In response to Appellants' arguments that the Examiner is dissecting claim features into a few words at a time and applying different references to these words in the abstract, the Examiner alleged that Appellants are arguing against the references individually instead of addressing the combination of references. Examiner's Answer, page 26.

Appellants appreciate the Examiner's dodging of the issue that remains here. As explained in the Appeal Brief, the Examiner has not provided any motivation for combining these alleged features of Hisgen et al. and Mattis et al. Therefore, the Examiner has not established a proper rejection under 35 U.S.C. § 103.

For at least the foregoing reasons and for those reasons presented in the Appeal Brief,

Appellants submit that the rejection of claim 17 under 35 U.S.C. § 103(a) based on Mattis et al. and Hisgen et al., whether taken alone or in any reasonable combination, is improper.

Accordingly, Appellants request that the rejection of claim 17 be reversed.

#### IV. CONCLUSION

In view of the foregoing arguments and the arguments presented in the Appeal Brief, Appellants respectfully solicit the Honorable Board to reverse the Examiner's rejections of claims 1-25 under 35 U.S.C. §§ 102 and 103.

To the extent necessary, a petition for an extension of time under 37 C.F.R. § 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account No. 50-1070 and please credit any excess fees to such deposit account.

Respectfully submitted,  
HARRITY SNYDER, LLP

/Paul A. Harrity, Reg. No. 39,574/  
Paul A. Harrity  
Reg. No. 39,574

Date: September 24, 2007  
11350 Random Hills Road  
Suite 600  
Fairfax, Virginia 22030  
(571) 432-0800